More Variables

1. Introduction

Today we continue to work with variables, loops, and conditional statements. We use these basic elements of any computer language to create increasingly sophisticated programs.

But these elements are ways to express steps in an algorithm. We need to devise the algorithm first. Devising an algorithm requires understanding the problem. Understanding the problem requires careful looking and thinking. The longer you look at a painting, the more you see. The more times you listen to a piece of music, the more you hear. The more times you read a book, the more you ‘see’. Problems are the same; you have to work at them, sketch, outline, plan. Jumping to code is like starting to type an English paper without thinking about the topic or making an outline.

2. Looking at Homework Problems

Let us ‘re-read’ the call-tree problem. Most people looked at each ‘tier’ of callers as taking so much time to get done. But is that really accurate. If you look closely, do you see something different? We talk about it, then watch a simulation. How does this re-view of the problem affect your design of the algorithm? Impatience is a virtue here.

Now let us re-view the card sorting problem. We look at a few solutions and look for loops. How can we be lazy and second involves repetition and changing sizes. Laziness is a virtue here.

3. Some Projects

Cow Clicks1 - After 10 clicks, shrinks then vanishes in a puff of smoke

Cow Clicks2 - Moos after each 10 clicks, no visible counting
Countdown and Runs Away (What repeats? What varies?)

algorithm code

Countdown and Accelerates Upwards (What repeats? What varies?)

algorithm code
Laws vs Radar Guns

```plaintext
set x position = 0
repeat until x position > 240
    say (x position)
    wait 3
    set (x position) = 300
    say (x position)
    wait 3
    change (x position) by -200
```

Project: moonbounce2

<table>
<thead>
<tr>
<th>variables:</th>
</tr>
</thead>
<tbody>
<tr>
<td>when flag clicked</td>
</tr>
<tr>
<td>go to x: (0) y: (0)</td>
</tr>
<tr>
<td>repeat (10)</td>
</tr>
<tr>
<td>repeat until (y position) &gt; 100</td>
</tr>
<tr>
<td>change y by 2</td>
</tr>
<tr>
<td>wait (0.005) secs</td>
</tr>
<tr>
<td>repeat until (y position) &lt; 1</td>
</tr>
<tr>
<td>change y by -2</td>
</tr>
<tr>
<td>wait (0.005) secs</td>
</tr>
<tr>
<td>play drum (48) for (0.2) beats</td>
</tr>
<tr>
<td>say [Ta-Dah!] for (2) secs</td>
</tr>
</tbody>
</table>

**How do we produce each of the following effects?**

1. Have the height of the bounce reduce by 5 units each bounce
2. Have each bounce be only half the height of the previous bounce
3. Have each bounce take 10% longer than the previous bounce
4. Make the bounce more realistic - cat slows as it nears the top
5. Play a random drum sound on each bounce
6. Create the illusion of the cat bouncing into the distance

**4. Using Variables to Draw Fancier Images**

This spiral is centered at the origin and it begins with a side of 300. The distance between the lines is 30 pixels. What loop with produce this image?